Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently amended) A method in a data processing system for gathering performance information associated with executing instructions, the method comprising:

executing instructions in an application by a processor in the data processing system;

detecting an indicator associated with an instruction in the executing instructions during execution of the instructions, wherein the indicator is a data retrieved from memory during execution of the instructions, and wherein the indicator specifies counting of events that are associated with execution of the instructions or counting of events that are associated with accesses to memory locations;

sending a signal indicating that the instruction associated with the indicator is being executed to a functional unit in the processor, wherein the functional unit in the processor counts events for instructions in the executing instructions associated with the indicators, and wherein events associated with instructions without indicators are not counted;

responsive to receiving the signal, counting, by [[a]] the functional unit of the processor, events that occur within the data processing system as specified by the indicator to form counted events, wherein the indicator enables counting of events on a per instruction or a per memory location basis, and wherein only events specified by indicators are counted;

retrieving a count value that represents the counted events to form the performance information; and

selecting, by the application, an execution path within a set of instructions based on the count value in the performance information gathered by the functional unit in the processor in accordance with the indicators associated with the executing instructions, wherein a branch instruction that selects the execution path is contained in an object code block that contains the instructions that were executed while detecting the indicator, and wherein the step of selecting an execution path further comprises:

executing instructions that determine if the count value satisfies a first condition; [[and]]

branching to execute a first set of instructions in response to a determination that the count value satisfies the first condition; and

branching to execute a second set of instructions in response to a determination that the count value satisfies a second condition.

2. (Currently amended) The method of claim 1 wherein counting events that occur within the data processing system as specified by the indicators to form the counted events further comprises:

identifying a threshold number of clock cycles needed to complete [[an]] the instruction associated with the indicator, wherein the threshold is set within the indicator; and

responsive to an event associated with executing the instruction associated with the indicator exceeding the threshold, counting the event.

3. (Currently amended) The method of claim 1 further comprising:

generating the indicators by a compiler ~~branching to execute a second set of instructions in response to a determination that the count value satisfies a second condition~~.

4. (Original) The method of claim 1 wherein the step of selecting an execution path further comprises:

calling one of a plurality of subroutines in response to a determination that the count value satisfies a specified condition, wherein each of the plurality of subroutines is a version of a given subroutine, wherein each version has been compiled differently from other versions of the given subroutine.

5. (Currently amended) A method in a data processing system for gathering performance information associated with executing instructions, the method comprising:

compiling source code statements, by a compiler, to generate instructions for a processor in the data processing system;

generating, by the compiler, indicators to be processed by the processor during execution of the instructions, wherein the indicators are data values that are retrieved from memory during execution of the instructions that specify counting of events that are associated with execution of the instructions or are data values that are retrieved from memory during execution of the instructions that specify counting of events that are associated with accesses to memory locations, and wherein the indicators specify counting of events on a per instruction or a per memory location basis such that only the events specified by the indicators are counted to form the performance information; and

generating instructions to select an execution path within a set of instructions based on a count value in the performance information that represents counted events, wherein a branch instruction that selects the execution path is contained in an object code block that contains the instructions that are executed while the indicators are processed.

6. (Currently amended) The method of claim 5 further comprising:

processing, by [[a]] the compiler, source code statements that control the selection of the execution path.

7. (Currently amended) The method of claim 5 further comprising:

determining, by [[a]] the compiler, alternative ways to compile the source code statements;

compiling the source code statements in multiple ways to generate multiple sets of instructions for the compiled source code statements;

and placing each set of instructions in a different execution path to be selected based on the count value.

8. (Original) The method of claim 7 further comprising:

identifying a compiler directive among the source code statements that specifies a type of autonomic monitoring to be implemented by the compiler; and

identifying alternative ways to compile the source code statements based on the compiler directive.

9. (Original) The method of claim 8 further comprising:

specifying an event in the compiler directive to be autonomically monitored.

10. (Currently amended) The method of claim 5 further comprising:

determining, by [[a]] the compiler, alternative ways to compile a subroutine within the source code statements;

generating a plurality of compiled versions of the subroutine, wherein each compiled version has been compiled differently from other compiled versions of the subroutine; and

placing the plurality of compiled versions of the subroutine in different execution paths to be selected based on the count value.

11. (Currently amended) The method of claim 5 further comprising:

processing, by [[a]] the compiler, at least one compiler directive that specifies compiler ~~directives that specify~~ the selection of the execution path.

12. (Currently amended) A computer program product on a computer readable recordable-type storage medium for use in a data processing system that executes instructions, the computer program product comprising:

~~means for executing instructions associated with an application by a processor in the data processing system;~~

~~means for~~ instructions for detecting indicators during execution of [[the]] instructions associated with an application, wherein the instructions are executed by a processor in the data processing system, wherein the indicators are data values that are retrieved from memory during execution of the instructions that specify counting of events that are associated with execution of the instructions or are data values that are retrieved from memory during execution of the instructions that specify counting of events that are associated with accesses to memory locations;

instructions for sending a signal indicating that the instruction associated with the indicator is being executed to a functional unit in the processor, wherein the functional unit in the processor counts events for instructions in the executing instructions associated with the indicators, and wherein events associated with instructions without indicators are not counted;

~~means for~~ instructions for counting, by [[a]] the functional unit associated with the processor, events that occur within the data processing system as specified by the indicators to form counted events, wherein the indicators enable counting of events on a per instruction or a per memory location basis, and wherein only the events specified by the indicators are counted;

~~means for~~ instructions for retrieving a count value that represents the counted events; and

~~means for~~ instructions for selecting, by the application, an execution path within a set of instructions based on the count value gathered by the functional unit in the processor in accordance with the indicators associated with the executing instructions, wherein a branch instruction that selects the execution path is contained in an object code block that contains the instructions that are executed while detecting the indicators, and wherein a first execution path is selected in response to a determination that the count value satisfies a first condition and wherein a second execution path is selected in response to a determination that the count value satisfies a second condition.

13. (Currently amended) The computer program product of claim 12 further comprising:

instructions for generating the indicators by a compiler ~~wherein the means for selecting an execution path further comprises:~~

~~means for executing instructions that determine if the count value satisfies a first condition; and~~

~~means for branching to execute a first set of instructions in response to a determination that the count value satisfies the first condition~~.

14. (Currently amended) The computer program product of claim 13 further comprising:

~~means for~~ instructions for branching to execute a second set of instructions in response to a determination that the count value satisfies a second condition.

15. (Currently amended) The computer program product of claim 12 wherein the means for selecting an execution path further comprises:

~~means for~~ instructions for calling one of a plurality of subroutines in response to a determination that the count value satisfies a specified condition, wherein each of the plurality of subroutines is a version of a given subroutine, wherein each version has been compiled differently from other versions of the given subroutine.

16. (Currently amended) A computer program product on a computer readable recordable –type storage medium for use in a data processing system that executes instructions, the computer program product comprising:

~~means for~~ instructions for compiling source code statements to generate instructions for a processor in the data processing system;

~~means for~~ instructions for generating, by the compiler, indicators to be processed, by the processor, during execution of the instructions, wherein the indicators are data values that are retrieved from memory during execution of the instructions that specify counting of events that are associated with execution of the instructions or are data values that are retrieved from memory during execution of the instructions that specify counting of events that are associated with accesses to memory locations, and wherein the indicators specify counting of events on a per instruction or a per memory location basis such that only the events specified by the indicators are counted; and

~~means for~~ instructions for generating instructions to select an execution path within a set of instructions based on a count value that represents counted events, wherein a branch instruction that selects the execution path is contained in an object code block that contains the instructions that are executed while the indicators are processed.

17. (Currently amended) The computer program product of claim 16 further comprising:

~~means for~~ instructions for processing, by [[a]] the compiler, source code statements that control the selection of the execution path.

18. (Currently amended) The computer program product of claim 16 further comprising:

~~means for~~ instructions for determining, by [[a]] the compiler, alternative ways to compile the source code statements;

~~means for~~ instructions for compiling the source code statements in multiple ways to generate multiple sets of instructions for the compiled source code statements; and

~~means for~~ instructions for placing each set of instructions in a different execution path to be selected based on the count value.

19. (Currently amended) The computer program product of claim 18 further comprising:

~~means for~~ instructions for identifying a compiler directive among the source code statements that specifies a type of autonomic monitoring to be implemented by the compiler; and

~~means for~~ instructions for identifying alternative ways to compile the source code statements based on the compiler directive.

20. (Currently amended) The computer program product of claim 19 further comprising:

~~means for~~ instructions for specifying an event in the compiler directive to be autonomically monitored.

21. (Currently amended) The computer program product of claim 16 further comprising:

~~means for~~ instructions for determining, by [[a]] the compiler, alternative ways to compile a subroutine within the source code statements;

~~means for~~ instructions for generating a plurality of compiled versions of the subroutine, wherein each compiled version has been compiled differently from other compiled versions of the subroutine; and

~~means for~~ instructions for placing the plurality of compiled versions of the subroutine in different execution paths to be selected based on the count value.

22. (Currently amended) The computer program product of claim 16 further comprising:

~~means for~~ instructions for processing, by [[a]] the compiler, compiler directives that specify the selection of the execution path.

23. (Currently amended) An apparatus in a data processing system that executes instructions, the apparatus comprising:

means for executing instructions associated with an application by a processor in the data processing system;

means for detecting indicators during execution of the instructions, wherein the indicators are data values that are retrieved from memory during execution of the instructions that specify counting of events that are associated with execution of the instructions or are data values that are retrieved from memory during execution of the instructions that specify counting of events that are associated with accesses to memory locations;

means for sending a signal indicating that the instruction associated with the indicator is being executed to a functional unit in the processor, wherein the functional unit in the processor counts events for instructions in the executing instructions associated with the indicators, and wherein events associated with instructions without indicators are not counted;

means for counting, by [[a]] the functional unit associated with the processor, events that occur within the data processing system as specified by the indicators to form counted events, wherein the indicators enable counting of events on a per instruction or a per memory location basis, and wherein only the events specified by the indicators are counted;

means for retrieving a count value that represents the counted events; and

means for selecting, by the application, an execution path within a set of instructions based on the count value gathered by the functional unit in the processor in accordance with the indicators associated with the executing instructions, wherein a branch instruction that selects the execution path is contained in an object code block that contains the instructions that are executed while detecting the indicators, and wherein a first execution path is selected in response to a determination that the count value satisfies a first condition and wherein a second execution path is selected in response to a determination that the count value satisfies a second condition.

24. (Currently amended) The apparatus of claim 23 further comprising:

a compiler, wherein the compiler generates the indicators ~~wherein the means for selecting an execution path further comprises:~~

~~means for executing instructions that determine if the count value satisfies a first condition; and~~

~~means for branching to execute a first set of instructions in response to a determination that the count value satisfies the first condition~~.

25. (Original) The apparatus of claim 24 further comprising:

means for branching to execute a second set of instructions in response to a determination that the count value satisfies a second condition.

26. (Original) The apparatus of claim 23 wherein the means for selecting an execution path further comprises:

means for calling one of a plurality of subroutines in response to a determination that the count value satisfies a specified condition, wherein each of the plurality of subroutines is a version of a given subroutine, wherein each version has been compiled differently from other versions of the given subroutine.

27. (Currently amended) An apparatus in a data processing system that executes instructions, the apparatus comprising:

a compiler, wherein the compiler compiles ~~means for compiling~~ source code statements to generate instructions for a processor in the data processing system[[;]] generates ~~means for generating~~ indicators to be processed during execution of the instructions, wherein the indicators are data values that are retrieved from memory during execution of the instructions that specify counting of events that are associated with execution of the instructions or are data values that are retrieved from memory during execution of the instructions that specify counting of events that are associated with accesses to memory location and wherein the indicators specify counting of events on a per instruction or a per memory location basis such that only the events specified by the indicators are counted to form the performance information; and ~~means for generating~~ generates instructions to select an execution path within a set of instructions based on a count value_that represents counted events, wherein a branch instruction that selects the execution path is contained in an object code block that contains the instructions that are executed while the indicators are processed.

28. (Currently amended) The apparatus of claim 27 ~~further comprising:~~

~~means for processing, by a compiler,~~ wherein the compiler processes source code statements that control the selection of the execution path.

29. (Currently amended) The apparatus of claim 27 further comprising: means for determining, by[[a]] the compiler, alternative ways to compile the source code statements;

means for compiling the source code statements in multiple ways to generate multiple sets of instructions for the compiled source code statements; and

means for placing each set of instructions in a different execution path to be selected based on the count value.

30. (Original) The apparatus of claim 29 further comprising:

means for identifying a compiler directive among the source code statements that specifies a type of autonomic monitoring to be implemented by the compiler; and

means for identifying alternative ways to compile the source code statements based on the compiler directive.

31. (Original) The apparatus of claim 30 further comprising:

means for specifying an event in the compiler directive to be autonomically monitored.

32. (Currently amended) The apparatus of claim 27 further comprising:

means for determining, by [[a]] the compiler, alternative ways to compile a subroutine within the source code statements;

means for generating a plurality of compiled versions of the subroutine, wherein each compiled version has been compiled differently from other compiled versions of the subroutine; and

means for placing the plurality of compiled versions of the subroutine in different execution paths to be selected based on the count value.

33. (Currently amended) The apparatus of claim 27 further comprising:

means for processing, by [[a]] the compiler, at least one compiler directive that specifies ~~compiler directives that specify~~ the selection of the execution path.